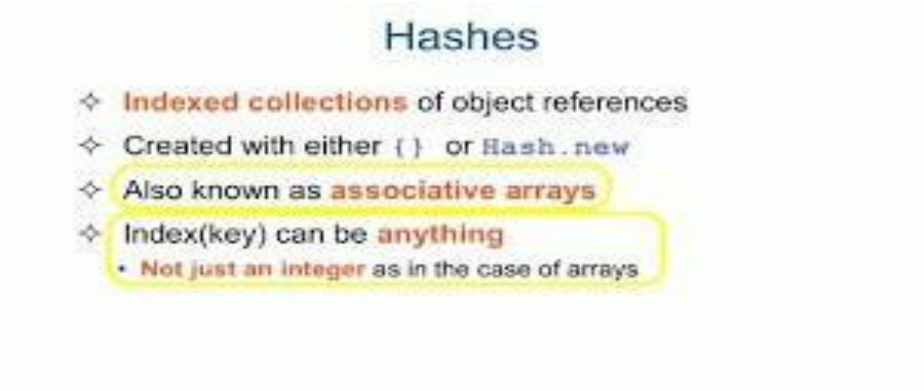
Ruby iterates over Hash

In this article, let us see What is Hash? and How Ruby iterates over Hash? and also a few examples of how to use hashes.

About Hash:

The data structure that saves objects by related keys, is known as Hash. Hash is different from an Array, that saves objects by an ordered index. Generally, the items in a hash are called key-value pairs, which makes a relative description of data.

If you want to Gain In-depth Knowledge on **Ruby**, please go through this link **Ruby On Rails Training**



Generally, We can make a hash by using data types as values and symbols as keys. In a hash, the key-value pairs are enclosed in curly braces { } and separated by commas.

We can create hashes in two ways. In first way, which is an older syntax. In this syntax, we use => symbol to filter the value and the key.

```
irb :001 > old_syntax_hash = {: name => 'john'}

=> {: name=>'john'}
```

Ruby version 1.9 introduced the second way i.e.,new syntax that is simple than the previous syntax

```
irb :002 > new_hash = {: name: 'john'}

=> {: name=>'john'}
```

By using two ways, we get the same results.

- We can also give more than one key-value pairs in hashes.

irb :003 > person = { weight: '60 kgs', height: '5.9 ft'}

=>{ weight: '60 kgs', height: '5.9 ft'}

- We can also add anything to a previous hash.

irb :004 > person[:hair] = 'black'

=> "black"

irb :005 > person

=> { :weight=> '60 kgs', :height=> '5.9 ft', :hair=>'black'}

irb :006> person[:age]=30

=>30

irb :007> person

=> { :weight=> '60 kgs', :height=> '5.9 ft', :hair=>'black', :age=>30}

- We can also delete anything from a previous hash

irb :008 > person.delete(:weight)

=> 60 kgs

irb 009 > person

=> { irb :005 > person

=> { :weight=> '60 kgs', :height=> '5.9 ft', :hair=>'black'}

- If you want to get any information from a hash.

irb :010> person[:age]

=>30

- We can get any information from a hash and also combine two hashes.

irb :011 > person.merge!(new_hash)

=>:height=> '5.9 ft', :hair=>'black', :age=>30, :name=>'John'}

How to do Iterating over Hashes?

More than one element can be there in hashes, so sometimes we can repeat a hash, again and again, to do something with every element. Iterating over hashes is the same as the iterating over arrays, but it has a small difference. In hashes, we will use 'each' method again.

Example:

```
# iterating_over_hashes.rb


person = {name: 'John', weight: '60 kgs', height: '5.9 ft', hair: 'black'}


person.each do |key, value|

puts "John's #{key} is #{value}"

end
```

We have assigned a variable to the value and key in 'each' method. In the above example, we have given the key to the 'key' variable and the value to the 'value' variable.

The output is:

```
John's name is John

John's weight is 60Kgs

John's height is 5.9 ft

John's hair is black
```

What are Common Hash Methods?

Let us see some common methods that we use in Ruby's Hash class.

1. **has_key?:** This method is used to, check if a hash has any specific key. A Boolean value is returned with this method.

2. **Select:** This method is used to, move a block and return any key-value pairs that test to true if it is run through the block.

3. **Fetch:** This method is used to, move a given key and returns a value of the key if it is there. We can also give a choice, for return if that key is not there.

4. **to_a:** When this method is called, it returns your hash in an array form. It will not change the hash completely.

5. **keys and values:** This method is used to, get all the values or the keys from the hash. It returns in the form of an array. You can do things like print all the keys in the hash, as it is giving an array.

If you are using an older version of Ruby i.e, Before Ruby 1.9, there is no need to give hashes in a specific order. But, if you are using the latest version Ruby 1.9, hashes are arranged in the order they are saved.

The key-value pair's concept is also well used in other technologies also, so it's better to have a grip on that topic. In this article, I have shared some information about What is hash? And how to iterate on the hash, I will share more information on Hashes in the next article. Enroll Live Free Demo On **Ruby On Rails Online Training**